Angular Advanced
Performance tips & tricks

Peter Kassenaar –
info@kassenaar.com

“Performance” has many faces

Build / load time performance

Run time performance

1. Tips on Load time performance - checklist

# *1.Optimize your builds* by using `ng build`

# 2. Use the `--prod` flag for:

AOT-compiling

Uglifying

Minifying

Removal of source maps

Bundling (by using WebPack)

Tree shaking (enabled by default)

And (much!) more

https://github.com/angular/angular-cli/wiki/build

#3 - Use *Lazy Loading* in your app

Don’t load anything that is not immediately necessary

At the very minimum use

`PreloadingStrategy: PreloadAllModules`

Consider writing a custom loading strategy

https://angular.io/guide/lazy-loading-ngmodules

#4 - Consider using *Server Sided Rendering* (SSR)

- Compiled app is served to the browser – fast startup time
- User interaction is captured and stored/cached until the complete app is loaded.
- Apps can be indexed, identified and analyzed by Google Bot
- Can be tricky to set up!
- Use Angular Universal module for SSR: https://angular.io/guide/universal

Jeff Whelpley on Angular Universal

https://www.youtube.com/watch?v=KiAnzAk04uA

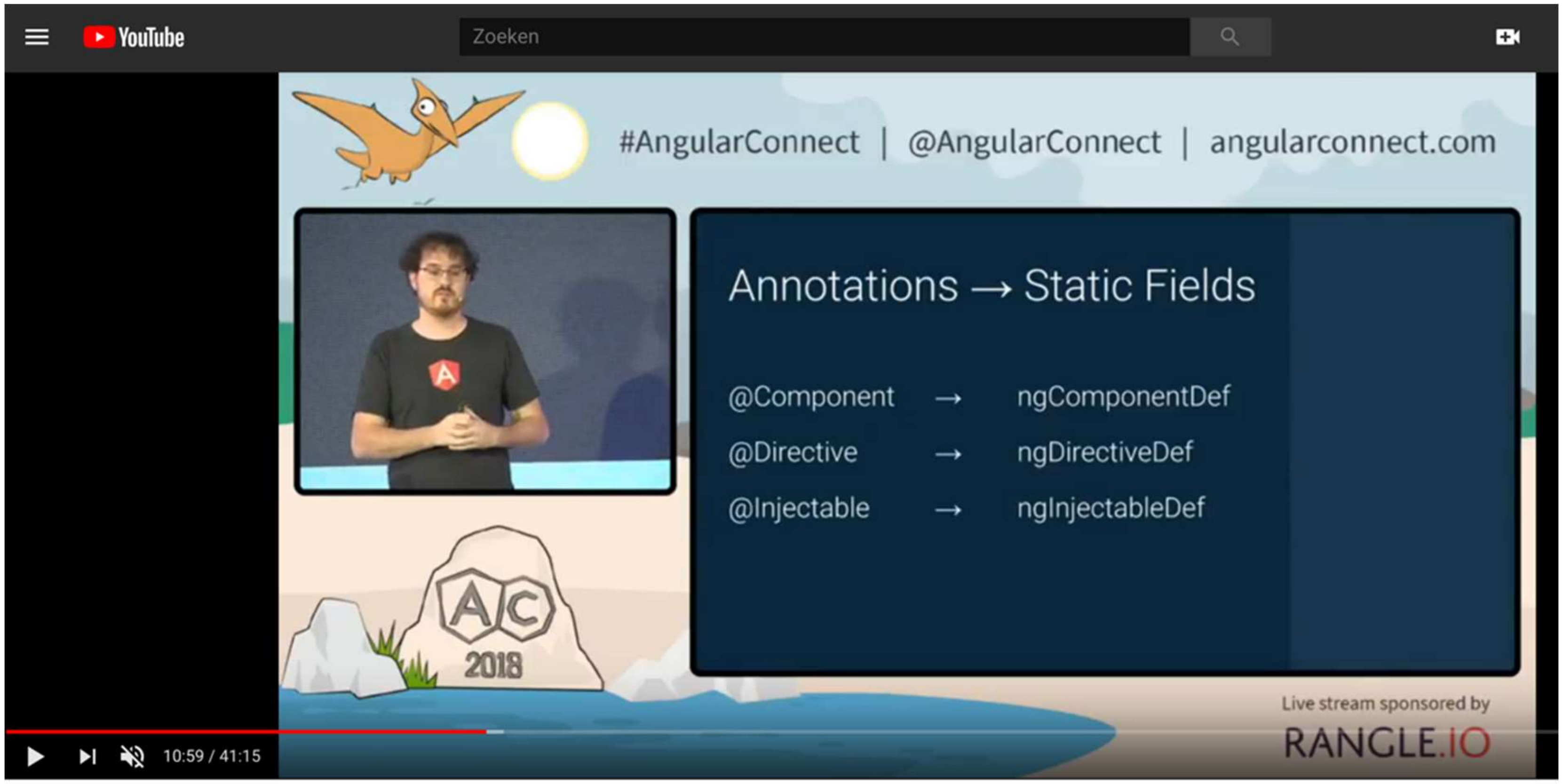
#5 - Update Angular CLI and Angular Packages regularly

- Newer builds typically provide smaller bundles, faster startup times etc.
- Ivy Renderer will be included by default

```
npm install -g @angular/cli

ng update
```

More on Angular Ivy Renderer



Annotations → Static Fields

@Component	→	ngComponentDef
@Directive	→	ngDirectiveDef
@Injectable	→	ngInjectableDef

The Theory of Angular Ivy | Alex Rickabaugh | AngularConnect 2018

Volgende

https://www.youtube.com/watch?v=isb5Ef6yI48

Docs: https://blog.nrwl.io/understanding-angular-ivy-incremental-dom-and-virtual-dom-243be844bf36

https://medium.com/js-imaginea/ivy-a-look-at-the-new-render-engine-for-angular-953bf3b4907a

On third party libs:

#6 - Use RxJS 6 or higher

Remove `rxjs-compat` when done!

#7 - Use a lib that is compatible with tree shaking

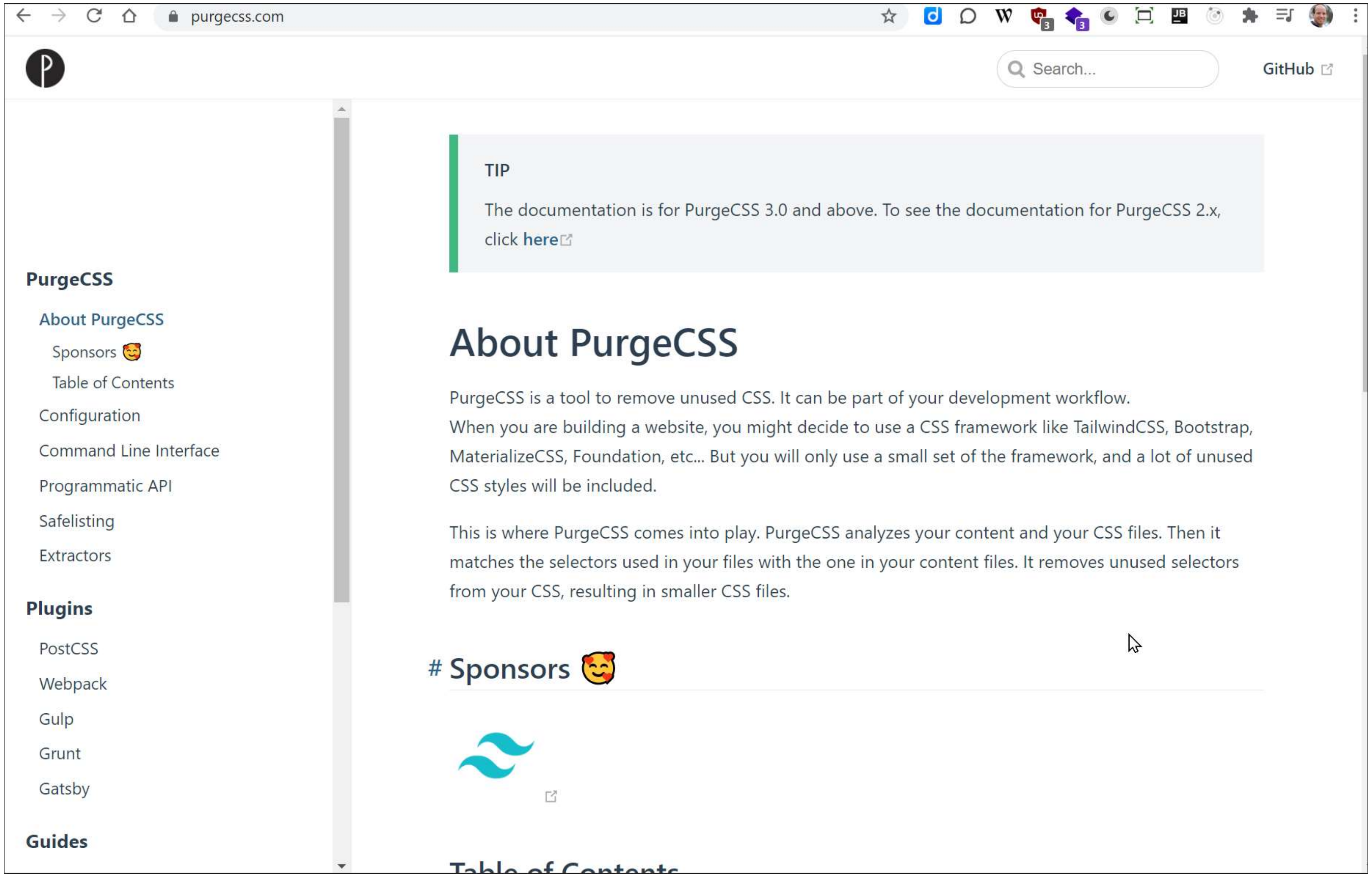
#8 - Don’t include everything. Only the stuff you need

i.e: create custom builds of Bootstrap, jQuery, Lodash, etc, if you decide to use these

#9 - Use vanilla JavaScript wherever possible.

- Often you don’t need lodash, jQuery anymore to perform basic tasks

PurgeCSS



PurgeCSS
About PurgeCSS
Sponsors 🥰
Table of Contents
Configuration
Command Line Interface
Programmatic API
Safelisting
Extractors

Plugins
PostCSS
Webpack
Gulp
Grunt
Gatsby

Guides

TIP

The documentation is for PurgeCSS 3.0 and above. To see the documentation for PurgeCSS 2.x, click here

About PurgeCSS

PurgeCSS is a tool to remove unused CSS. It can be part of your development workflow.
When you are building a website, you might decide to use a CSS framework like TailwindCSS, Bootstrap, MaterializeCSS, Foundation, etc... But you will only use a small set of the framework, and a lot of unused CSS styles will be included.

This is where PurgeCSS comes into play. PurgeCSS analyzes your content and your CSS files. Then it matches the selectors used in your files with the one in your content files. It removes unused selectors from your CSS, resulting in smaller CSS files.

Sponsors 🥰

https://purgecss.com/

#10 - Use gzip compression on your backend!

Compress the files on the server

#11 – Use H5bp server configurations (nginx, Apache and more - https://h5bp.org/

#12 - Use Http/2 where possible

Not possible if you need to support <= IE11

#13 - Compress your images

Consider using a tool like TinyPNG to compress images from your IDE

https://marketplace.visualstudio.com/items?itemName=andi1984.tinypng

Other image compression tools are available:

https://www.google.nl/search?q=image+compressor

#14 - Remove unused fonts from the app

Remove <style> hyperlinks from <head>

Remove unused fonts from /fonts directory

2 - Runtime performance - checklist

#15 - Use `ChangeDetection.OnPush` to avoid unneccessary evaluation of component trees

> This is the #1 runtime performance tip. Often overlooked!

#16 - Detach the Change Detector completely if you want full control over CD

> `this.cdr.detach` in `ngAfterViewInit() { … }`
>
> `this.cdr.detectChanges()` when you want to perform CD on demand.

#17 - Use `trackBy: trackbyFn` in your `*ngFor`-loops

- https://netbasal.com/angular-2-improve-performance-with-trackby-cc147b5104e5
- https://angular.io/api/common/NgForOf#ngForTrackBy
- Avoid expensive DOM-operations

#18 - Use *pipes* to format stuff in the UI.

Don’t let CD handle this (as this can become very expensive quite fast!)

https://codeburst.io/angular-tips-the-importance-of-pipes-49be3b1e99e7

#19 - Don’t do computations in the View/UI

DOM is slow

Use TypeScript for that

#20 - Remember to *unscubscribe* your observables to avoid memory leaks

Or let Angular `async` pipe handle that for you

#21 - If you have multiple subscribers to a source, use the `share()` operator

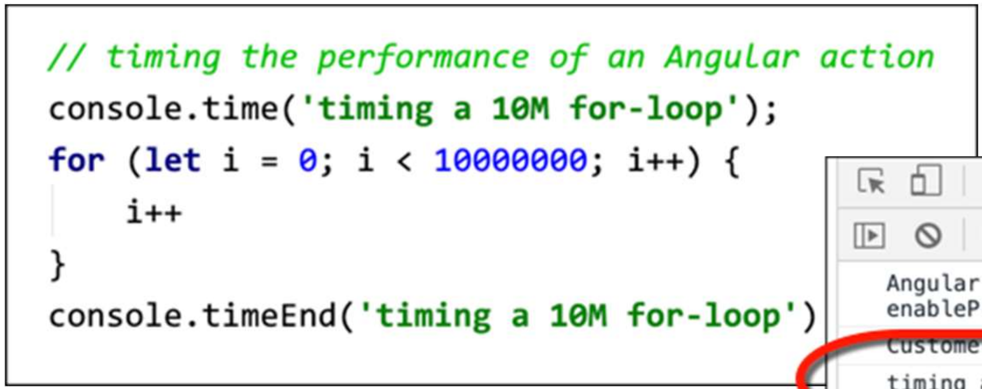
This avoids the processing of duplicate data among subscribers.

```
this.http.get<any>(‘http://some/endpoint').pipe(share());
```

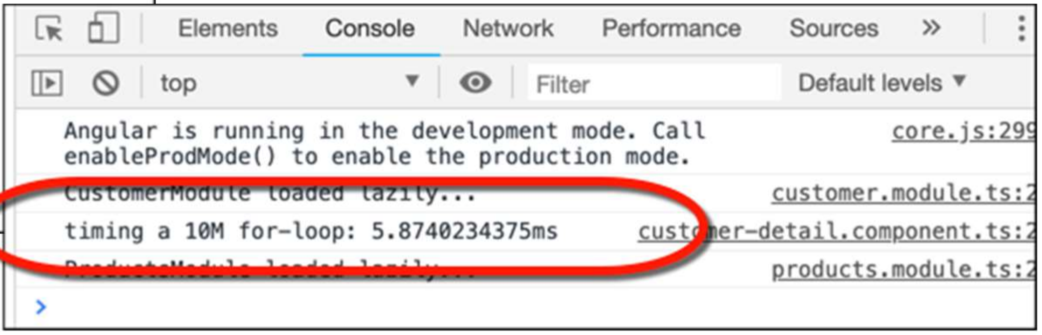
Q: "How to measuring response times for angular actions?"

A: You can use `console.time()` for that

https://alligator.io/js/console-time-timeend/



```
// timing the performance of an Angular action
console.time('timing a 10M for-loop');
for (let i = 0; i < 10000000; i++) {
    i++
}
console.timeEnd('timing a 10M for-loop')
```

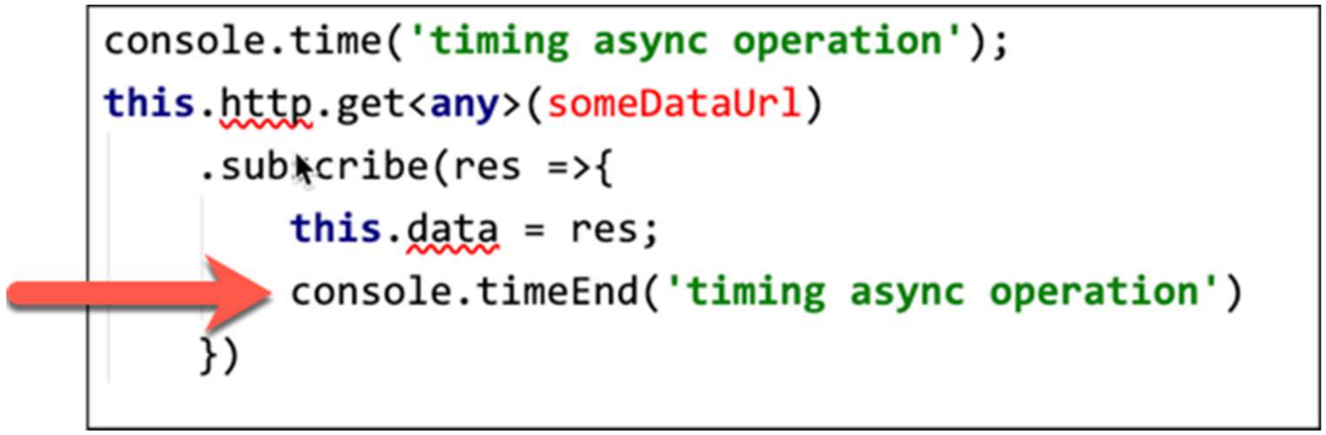


```
Angular is running in the development mode. Call          core.js:29
enableProdMode() to enable the production mode.
CustomerModule loaded lazily...                        customer.module.ts:2
timing a 10M for-loop: 5.8740234375ms       customer-detail.component.ts:2
ProductsModule loaded lazily...                        products.module.ts:2
```

Timing async operations

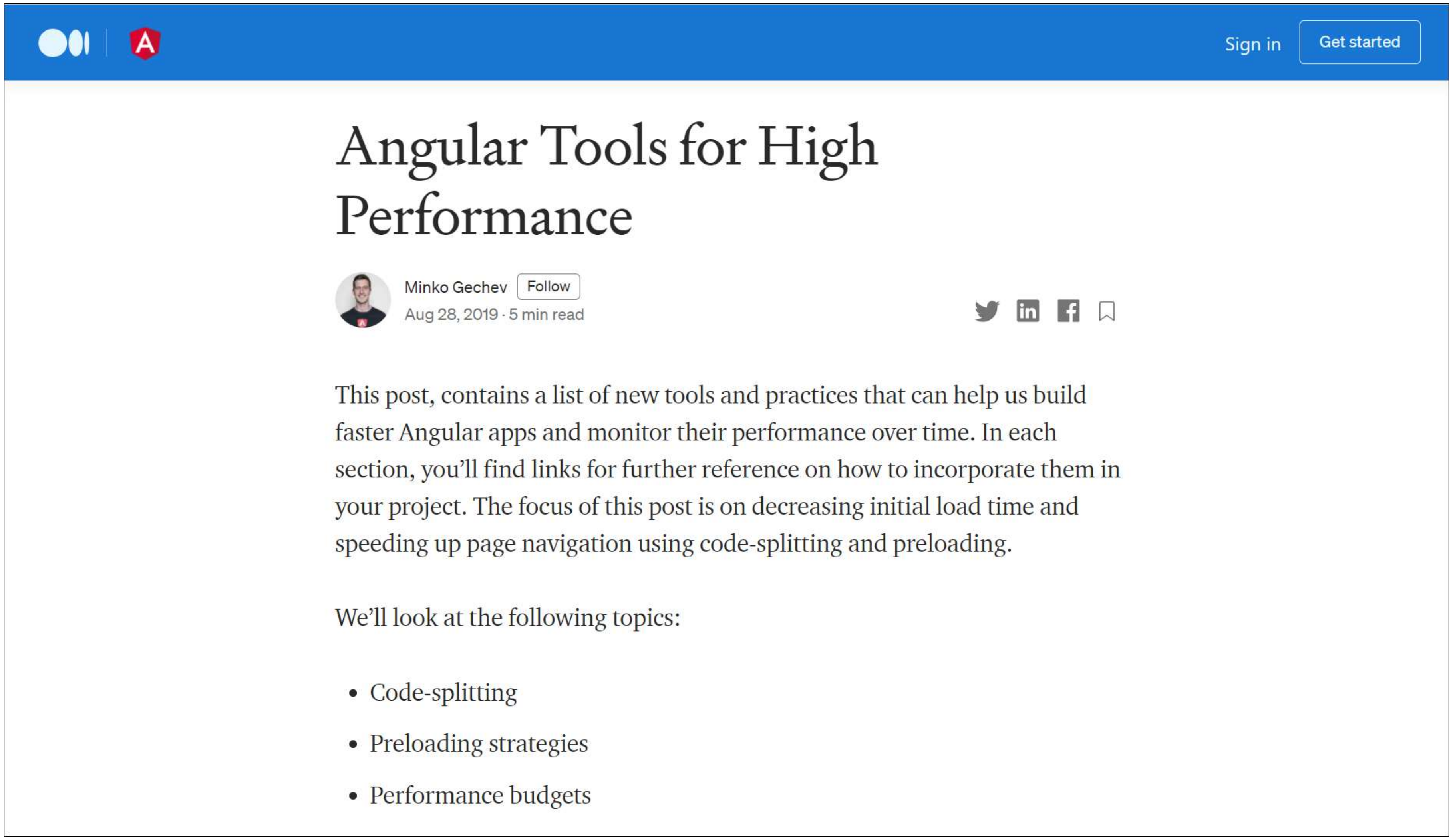
Beware - when using an async operation, be sure to place the `console.timeEnd()` *inside* the callback.

Not right after it!



```
console.time('timing async operation');
this.http.get<any>(someDataUrl)
    .subscribe(res =>{
        this.data = res;
        console.timeEnd('timing async operation')
    })
```

Minko Gechev – lots of articles/videos



Angular Tools for High Performance

Minko Gechev Follow
Aug 28, 2019 · 5 min read

This post, contains a list of new tools and practices that can help us build faster Angular apps and monitor their performance over time. In each section, you'll find links for further reference on how to incorporate them in your project. The focus of this post is on decreasing initial load time and speeding up page navigation using code-splitting and preloading.

We'll look at the following topics:

- Code-splitting
- Preloading strategies
- Performance budgets

QuickLinks & Predictive Prefetching

For larger apps, we can apply more advanced preloading heuristics:

Quicklink — preload only modules associated with visible links in the viewport

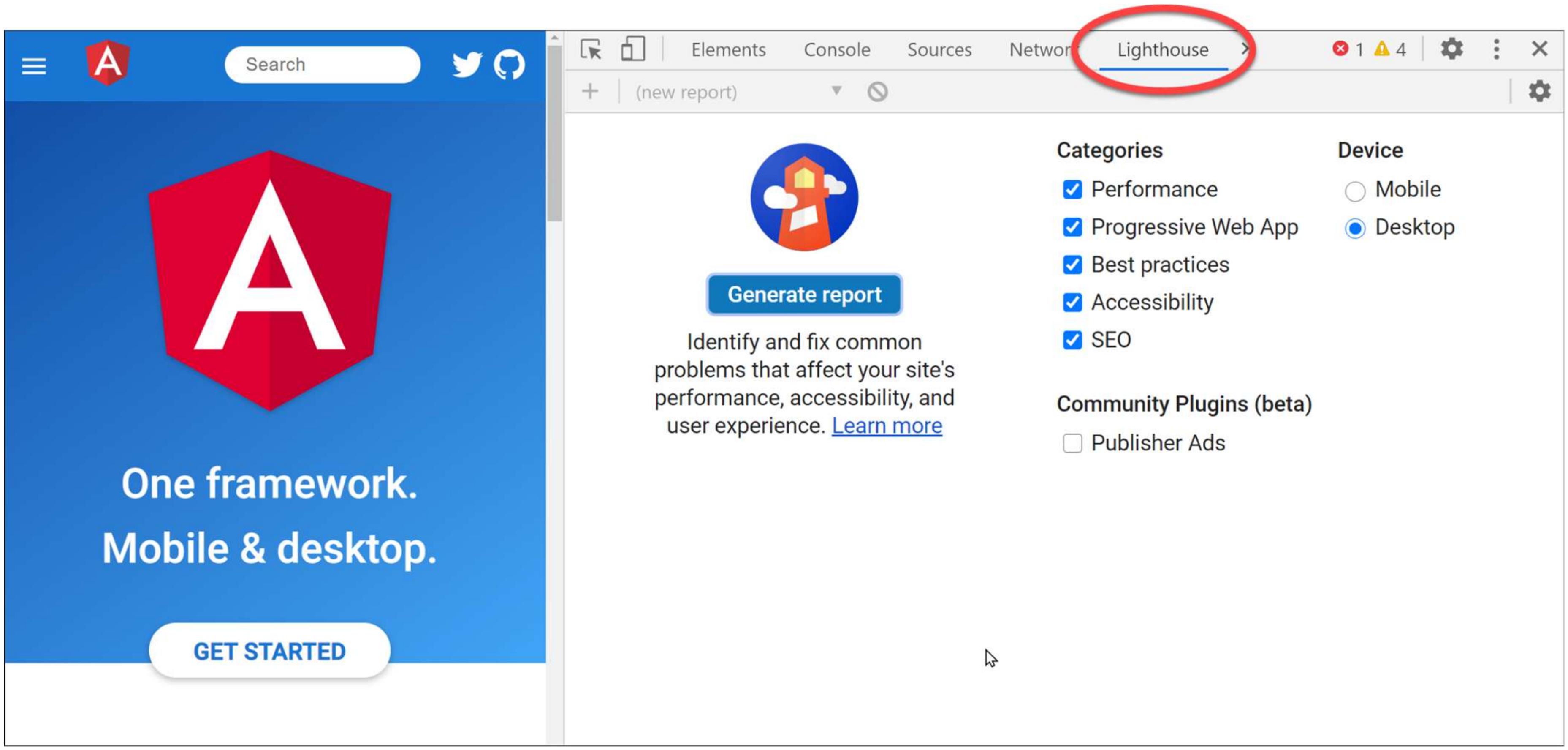
www.npmjs.com/package/ngx-quicklink

Predictive prefetching — preload only the modules that are likely to be needed next

github.com/guess-js/guess

www.youtube.com/watch?v=5FRxQiGqqmM

Use Lighthouse – Chrome DevTools



Identify and fix common problems that affect your site's performance, accessibility, and user experience. Learn more

Categories
- ✓ Performance
- ✓ Progressive Web App
- ✓ Best practices
- ✓ Accessibility
- ✓ SEO

Device
- ○ Mobile
- ● Desktop

Community Plugins (beta)
- ☐ Publisher Ads

10 tricks to optimize your Angular App

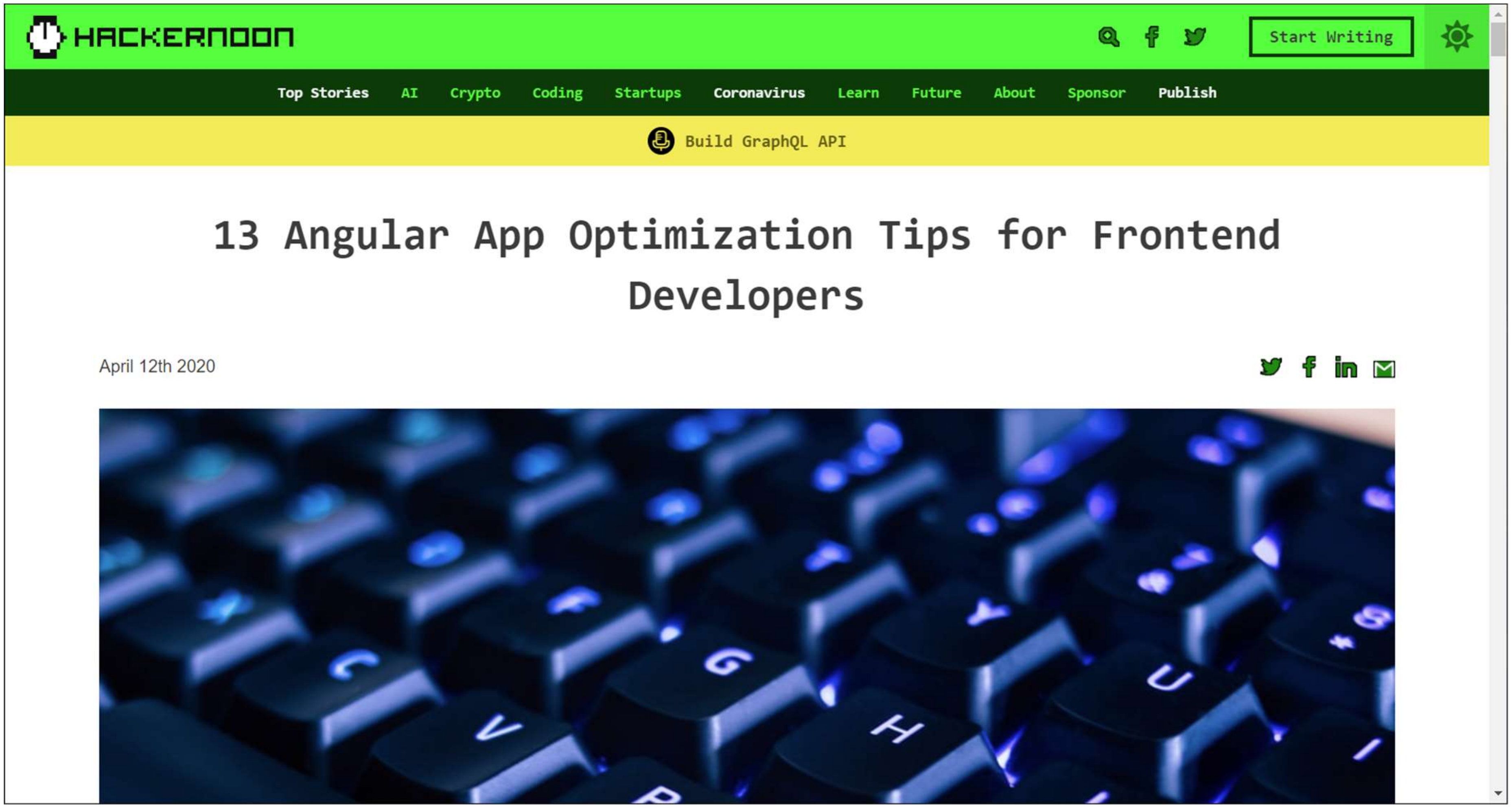
10 Tricks to Optimize Your Angular App

Chidume Nnamdi 🔥💻🎵🎮

Dec 24, 2019 · 12 min read

https://blog.bitsrc.io/10-tricks-to-optimize-your-angular-app-44208f616bf0

13 Angular App optimization tips



13 Angular App Optimization Tips for Frontend Developers

April 12th 2020

https://hackernoon.com/13-angular-app-optimization-tips-for-frontend-developers-z392329t

More info

- https://blog.thoughtram.io/angular/2017/02/02/making-your-angular-app-fast.html
- https://www.youtube.com/watch?v=ybNj-id0kjY – Minko Gechev –Optimizing an Angular application
- https://github.com/mgechev/angular-performance-checklist
- https://medium.com/@spp020/44-quick-tips-to-fine-tune-angular-performance-9f5768f5d945